班级: 2021 整合科学班 学号: 2000012272 姓名: 张广欣 评分:

同组学生: 无

实验名称: Labview 入门

1 埃氏筛

(附带存成.csv 格式的部分)

1. 原理

给出 n, 找出 \sqrt{n} 以内的素数 $p_1, p_2, ..., p_k$ 。先用 2 去筛,把 2 留下,把 2 的倍数剔除掉;再用下一个素数,也就是 3 筛,把 3 留下,把 3 的倍数剔除掉;接下去用下一个素数 5 筛,把 5 留下,把 5 的倍数剔除掉;不断重复。

算法复杂度: O(NloglogN)

2.python 实现

```
def eratosthenes(n):
filter = [1]*(n+1)
for i in range(2, int(n ** 0.5) + 1):
    if filter[i] == 1:
        for j in range(i * i, n + 1, i):
            filter[j] = 0
return [x for x in range(2, n + 1) if filter[x] == 1]
```

3. 运行结果(此结果为录制视频中的)

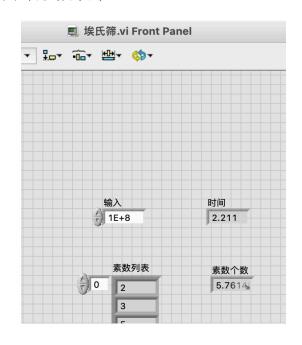


图 1: 埃氏筛运行时间(约 2.2 秒)

4. 一些发现

(1) Labview 中各个函数的实现时长差异巨大

最初我采用的是在空数组后每次加入新的素数,此方法理论上也只有 O(1) 的复杂度,但实际上此时的程序的运行时间接近 60 秒。之后改成先生成给定大小的空素数表,再将其中的元素替换成得到的素数,速度大幅度提升。

(2) Labview 版本的兼容性不佳

我用的是 Mac21 版的 Labview,该版本的.vi 脚本无法在 20 版的软件上运行,将脚本降级成 20 版的程序后,用 20 版 labview 打开时常常会出现文字乱码,有的电脑运行时间比我的(Macbook Pro 2019) 长 6 倍以上。可能是版本之间存在重新编译存在等问题。

最终版本的埃氏筛在另一台 20 版 Labview 上测试正常(甚至更快),但不一定每一台电脑都可以用 20 版的脚本复现出 2.2s 的速度(当然也可能更快)。因此录制视频作为证明。

2 欧拉筛

(对应的.vi 文件仅是实现欧拉筛的思路,但没有优化具体的函数实现以及添加存储模块,仅展示其原理,评分时请参考埃氏筛。)

1. 原理

任一合数只被:

"最小质因数 × 最大因数(非自己)= 这个合数"的途径删掉。

正确性证明:

设一合数 C 的最小质因数是 p_1 , 令 $B=C/p_1$,则 B 的最小质因数不小于 p_1 (否则 C 也有这个更小因子)。那么当外层枚举到 num = B 时,我们将会从小到大枚举各个质数;因为 num = B 的最小质因数不小于 p_1 ,所以 num 在质数枚举至 p_1 之前一定不会 break,C 一定会被 $B \times p_1$ 删 去。

当然质数不能表示成"大于1的某数×质数",所以整个流程中不会标记。

时间复杂度: O(n)

2.python 实现

```
def Euler (upperBound):
filter=[False for i in range(upperBound+1)]
primeNumbers=[]
for num in range(2,upperBound+1):
    if not filter[num]:
        primeNumbers.append(num)
    for prime in primeNumbers:
        if num*prime>upperBound:
             break
        filter[num*prime]=True
```

if num%prime == 0:

break;

return primeNumbers

3. 运行结果

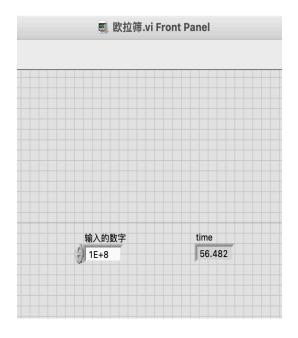


图 2: 欧拉筛运行时间

此时没有对运行函数进行优化,仅验证其正确性。若采用更优的函数实现,会比埃氏筛快一点。参考文献:

[1] Wiki. Sieve of Eratosthenes [EB/OL]. [2022.5.19]. https://en.wikipedia.org/wiki/Sieve_ $of_Eratosthenes$

[2] 程画. 欧拉筛法 [EB/OL]. [2022.5.19]. https://blog.csdn.net/chczy1/article/details/80327323.